

Semantic Fisheye Views

David Le Comte
Honours Student
University of Canterbury
Christchurch
New Zealand

November 7, 1997

Abstract

Visualisation of large quantities of information requires distortion for retention of context and detail. *Fisheye views* are based on fisheye lenses, which retain detail while introducing greater context. Typically this form of distortion compacts more information into a smaller region, irrespective of its relevance. Retaining irrelevant information clutters screen space, causing information overload which overwhelms a users cognition. Distorting the information based on its underlying structure, allows for selective suppression, producing a more meaningful display with less content. Typically the underlying structure can be represented by a graph, consisting of vertices and edges. This structure takes many forms, in this paper the limitations to these forms are discussed and extensions are presented which allow greater flexibility, retaining both detail and context while reducing information overload.

Thank You

Thanks to my supervisor Dr. Neville Churcher for introducing and motivating my interest in distortion oriented views. Also thanks to my partner Mikaela for her patience and support.

Contents

1	Introduction	5
2	Visualisation Techniques	7
2.1	Virtual Screens	7
2.2	Magnification Lenses	7
2.3	Overview Windows	8
2.4	Fisheye Views	8
2.4.1	Structure Versus Appearance	9
2.4.2	Validation	11
2.4.3	Thresholds	12
2.4.4	Outlines	13
3	AquaView: An Evaluation Tool	15
3.1	An Example	15
4	Structural Techniques	18
4.1	Some Terminology	18
4.2	Tree Structures	19
4.2.1	Extending the flexibility	19
4.2.2	Multiple Foci	21
4.3	Directed Acyclic Graphs	24
4.3.1	Extending the API	24
4.3.2	Alternative Distance Measures	25
4.4	Cyclic Digraphs and General Graphs	27
5	Conclusion	30
5.1	Future Work	30
5.2	Summary	30
A	Proving the Ordinality of Equation 4.2	33
A.1	Ensuring Focus has Highest DOI	33
A.2	Ensuring Root Path Nodes have Second Highest DOI	33
B	A Tree Analogous All Pairs Shortest Paths Algorithm	35
C	The Directory Outline	37

List of Figures

1.1	University of Canterbury as viewed through a fisheye lens	5
2.1	Alternative magnified views for xdvi	8
2.2	Appearance versus structural based distortion of text	10
2.3	Magnification regions for the Bifocal Display	10
2.4	Possible outlines for an early version of this report	13
3.1	A screen shot of AquaView	16
3.2	Screen shots of <code>filemgr</code>	17
4.1	Visual comparison of functional forms for tree structures	20
4.2	Multiple foci views	23
4.3	A DAG with two roots and a diamond	24
4.4	Comparison of distance measures for multiple roots.	26
4.5	Distance methods for acyclic digraphs	28
4.6	A cyclic digraph	29

List of Tables

2.1	Results from Furnas'	11
2.2	Results from Keahey and Marley	11
2.3	Results from Schaffer <i>et al.</i>	12
4.1	Thresholds used in figure 4.2	24
4.2	Comparison of functional forms for multiple roots	25

Chapter 1

Introduction

During visualisation, people perceive both detail and global context, surrounding a point of reference. While such visualisation is taken for granted in the real world, simulation of such is difficult in the electronic world, due to the limitations in screen size. Conventional methods: scrolling; zooming; and multiple windows, are inadequate for retention of detail and context without displaying a colossal amount of information.

A relatively new visualisation technique has been proposed, which distorts the visual image of the information. The most prolific is based on magnification of the information, where varying levels of detail are displayed. Magnified views far surpass their physical counterpart, allowing for a wider range of magnification and visualisation enhancements over localised areas. A new technique, which also distorts the information space, and based on the fisheye lens concept, has motivated active research into distortion techniques. Fisheye views strive for retention of both global context and high detail around a point of reference.

In figure 1.1, a photo of the University of Canterbury is shown as seen through a fisheye lens. Detail near the focus, (the buildings of the campus), are shown clearly, while an outline of the city is still visible. This high detail/global context retention is the basis for the fisheye views that are being developed for visualisation of large computerised information bases.

Distortion on the visual appearance of the information is the standard technique supported. The information connection is explicitly stated by the relative proximity of each information “point.” Results in favour of these techniques are not definitive, although this form of visualisation is in the early stages of development, so the possibilities may still be realised. Appearance based techniques attempt to fit more information on the screen as the context, which creates information overload, disturbing a users cognitive processes.

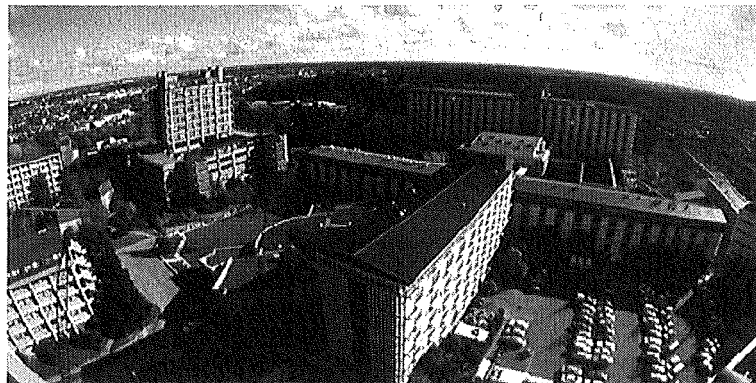


Figure 1.1: University of Canterbury as viewed through a fisheye lens

An alternative form of distortion is based on the underlying information structure, which promises a greater semantic representation of the information. The structure is typically represented by a graph, containing vertices and edges. Research into distortion on this graph structure has mainly been limited to tree forms. In this paper, the flexibility of tree structures is extended, (section 4.2), and a discussion on techniques for a more general class of graphs is made, (section 4.3). In chapter 2 different visualisation techniques are discussed, with the emphasis on *fish-eye views*.

Chapter 2

Visualisation Techniques

Visualising information is hampered by the need for tools which allow for local detail, (information near the user’s point of interest), while retaining global context, (an outline and relative positioning within the whole). Traditional visualisation techniques: scrolling and zooming, trade off one component for the other. As the available screen space is significantly smaller than the size of the information, only partial visualisation is possible. This chapter discusses techniques that are adopted, for simulating expansion of the screen space.

2.1 Virtual Screens

The concept of a virtual screen is a well known metaphor for displaying large quantities of information. The physical window is used as a viewer into the virtual window. That is, a portion of the information is displayed on the physical window with the remainder available by *moving*, (scrolling), either the physical window over the virtual window, or the virtual window under the physical window.

This approach allows for the display of a fixed level of detail only, and shows no global context. Of course in conjunction with alternate techniques, (discussed below), context is increased.

Scrolled views remove the peripheral information available to users, creating a form of tunnel vision. Tunnel vision is, generally, a negative attribute, handicapping a users perception. Simulation of the peripheral region motivates the techniques discussed in the rest of this chapter.

2.2 Magnification Lenses

While virtual screens allow for a single level of detail, an analogy to magnifying lenses is used to enable differing levels of detail. This technique “zooms the information” in, giving a more detailed view, or out displaying greater context. When the full context is displayed the detail is too small to work with. Zooming the view in which increases the detail, necessarily reduces the context.

Magnification techniques exist which are far superior to the magnifying glass analogy in which they were based. Bier *et al.*, [2] discusses a taxonomy of see-through tools which enable a distortion of the information, that far exceeds the capabilities of the *magnifying lens*. These tools allow for different magnifications to be applied to different regions of the display. This form of localised magnification, allows for display of a simultaneous context and detail view.

Keahey and Robertson [13] discuss the traditional technique used by the dvi viewer, `xdvi`.¹ They compare the occlusion problem that the `xdvi` magnification has with a non-linear magnification technique. The non-linear technique shows great detail while retaining the global overview, and is a non-occluding view of the information. Figure 2.1 shows the two views. Figure 2.1(a)

¹`xdvi` is an X Windows application for previewing DeVice Independent files, such as are produced by `TEX`.

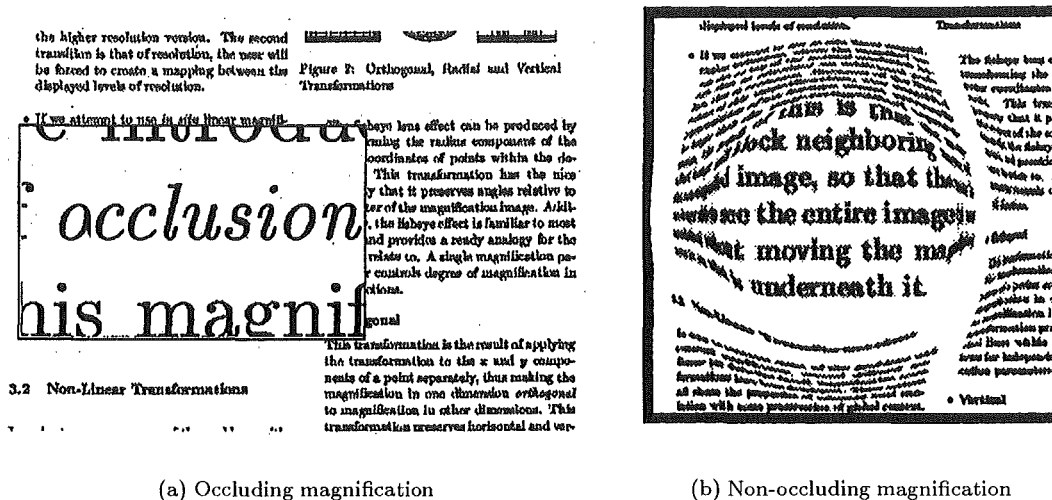


Figure 2.1: Alternative magnified views for xdvi

shows magnified detail, yet the magnification region is so large that information is covered, diminishing the local detail. The non-occluding display in figure 2.1(b), gives a highly detailed view of the users focal region, while retaining the global context. This non-occluding magnified view can be classed as an appearance based or graphical (Leung and Apperley [17]) fisheye view, which is discussed in section 2.4.

2.3 Overview Windows

Realisation that the screen dimension of a single window was insufficient for displaying the information, multiple windowing was introduced. Multiple windowing utilises separate windows for detail and context.

This problem suffers from the same problem that motivated its creation, namely there is insufficient display space for the multiple windows. This is analogous to the occluding display for *xdvi*'s magnification, technique on a large scale. Windows overlap, making it harder to integrate the context and detail. As the windows occlude each other, locating the windows becomes a problem. This problem is being alleviated by icon windows, which are shortcuts for locating windows. These icon windows introduce more screen clutter, and are a form of global context viewer. The integration between window and its icon window representation, must be made manually.

The problem of manual integration of the context and detail is a major limiting factor of this windowing technique. The final technique removes this manual cognition problem.

2.4 Fisheye Views

In 1986, Furnas [6] introduced generalised fisheye views as an alternate viewing strategy for electronic information. He described a general *degree of interest*, (DOI), function for an information point. A point can be arbitrarily defined, depending on the appropriate granularity required. A geographical example may define a point as cities and towns, or as provinces and countries.

Revisiting figure 2.1(b), text near the focus is detailed and as the distance increases the detail diminishes. This is the basic requirement of fisheye views and is a graphical equivalent to the

distortion given by the fisheye lens in figure 1.1 — where an information point was defined as a screen pixel.

In any distortion the relative sizes of points are retained in the distorted view. As the distance from the focus increases the size of the point decreases. This observation results in defining the DOI function, as consisting of two components: the *a priori interest*, (API), of a point; and the *distance* between the point and the focus. The DOI function can be regarded as a measure of how interesting a point is given the current focus and *world state* — in a different world state the interest of a point may change. Formally, the degree of interest of a point x given a focal point f , is defined as:

$$DOI(x|f) = API(x) - distance(x, f) \quad (2.1)$$

The API of a point is its self importance, or its *size*. The importance of a point is generally a subjective measure, although in the physical view it is most likely related to the screen size of the point. As an example, when a person subjectively assigns *a priori* values to towns and cities, in their country of origin, their place of residence will typically have a higher importance than a place they have never visited.

Distance can also be arbitrarily defined, Euclidean distance, which is the length of the straight line connecting two points, is not the only distance measure that can be used. In section 4, the distance is based on the distance between two nodes in an abstract graph.

The DOI equation is a generalisation or abstraction of the fisheye process. The n most interesting points correspond to the points with the n highest DOI values. This allows for the idea of thresholding, or suppression, of less interesting information. A threshold value, t , is chosen and all points which have a DOI greater than or equal to t are shown, the rest are suppressed. Generally, suppression of information is either removal from the display, or a reduction in size. In section 2.4.3 the consequences of this approach will be discussed more fully.

2.4.1 Structure Versus Appearance

There are two approaches for distorting views: structural and appearance based. In a structural view there is an implicit relationship between the information, whereas in an appearance based view there is an explicit relationship between the information. In figure 2.2, two distortion oriented views of text are shown. Figure 2.2(a) is an appearance based distortion, and figure 2.2(b) is a structural distortion.

In an appearance based approach the information is distorted according to its physical appearance. This approach is influenced by the size and proximity of the information points, hence the explicit relationships, and is a graphical rendering equivalent to the view through a fisheye lens. There has been an intensive study on techniques based on this form of distortion, Leung and Apperley [17] present an unification and taxonomy which describes a number of such techniques. These approaches attempt to fit more information onto the screen, by magnifying and shrinking different portions of the display. A restriction similar to data compression methods applies, namely, on average, the overall magnification is zero [32]. For every point that is magnified there is an equivalent compression in another part of the display. Newton's third law [10] summarises this nicely:

“For every action, (expansion) there is an equal and opposite reaction, (compression).”

This push/pull method of distortion can be summarised by a description of the Bifocal Display method [17] in figure 2.3. The bifocal display method simulates depth in the demagnification regions. This depth allows for increased information to be displayed, albeit at a decreased detail level. This form of distortion is a Cartesian distortion, which, as Sarkar and Brown [23] noted, was an unnatural distortion of the information space. They proposed an alternative method based on a polar transformation. A polar transformation maps the information onto a visually curved domain, whereas a Cartesian transformation is a mapping onto a step like structure.

The underlying structure of information can be extracted by an appropriate outlining tool, where the outline extracted will be in the form of parent-child relationships. The conjunction

```

24 {t[k-1] %= 10000;
25 k++;
26 }
27 case 'e':
28   t[0] = (t[0]+10000)
29   - x[0];
30   for (i=1; i<k; i++){
31     t[i] = (t[i]+10000)
32     - x[i];
33   }
34   t[k-1] %= 10000;
35   break;
36 case 'q':
37   for(i=0; i<k; i++) t[i] = x[i];
38   break;
39 case 'q':
40   exit(0);
41 default:
42   noprint = 1;
43   break;
44 }
45 if (!noprint){
46   for(i=k-1; i>0; i--){
47     printf("%d", t[i]);
48     if (i%9 == 0) printf("\n");
49     if (i%9 != 0) printf(" ");
50   }
51   printf("\n");
52   for(i=0; i<k; i++) x[i] = 0;
53 }
54 }
55 }
56 }

```

```

1 #define DIG 40
2 #include <stdio.h>
3 main()
4 {
5   int c, i, x[DIG/4], t[DIG/4], k = DIG/4, noprint = 0;
6   while ((c=getchar()) != EOF) {
7     if (c >= '0' && c <= '9') {
8       } else {
9         switch(c) {
10           case '+':
11             case '-':
12             case 'e':
13               for(i=0; i<k; i++) t[i] = x[i];
14               break;
15             case 'q':
16               default:
17                 noprint = 1;
18                 break;
19           }
20         if (!noprint){
21           for(i=k-1; i>0; i--){
22             printf("%d", t[i]);
23             if (i%9 == 0) printf("\n");
24             if (i%9 != 0) printf(" ");
25           }
26           printf("\n");
27           for(i=0; i<k; i++) x[i] = 0;
28         }
29       }
30     }
31   }
32 }

```

(a) Appearance based

(b) Structure based

Figure 2.2: Appearance versus structural based distortion of text

De-magnification in both X and Y dimensions	De-magnification in Y dimension	De-magnification in both X and Y dimensions
De-magnification in X dimension	Central Focus Region no de-magnification	De-magnification in X dimension
De-magnification in both X and Y dimensions	De-magnification in Y dimension	De-magnification in both X and Y dimensions

Figure 2.3: Magnification regions for the Bifocal Display

of these parent-child relationships can be used to build an arbitrary graph. In the structural distortion, information is suppressed, or “cut-off”, as defined by the threshold values, whereas the information is transformed in the visible approach. Suppression reduces the information that is being displayed on the screen, while retaining context and detail.

In figure 2.2, two distorted views of the same text² are displayed. The appearance based distortion reduces the size of the information points using font transformations. An information point is set to a line of text in the appearance based view. In contrast the structural view distorts

²This is a distortion of the program code presented by Furnas [6] in his paper on “Generalised Fisheye Views.”

the information based on the underlying structure of the code. The structure is represented as a tree, (trees are discussed section 4.2), which is based on the indentation of the statements.

2.4.2 Validation

In this section the validation of fisheye techniques is examined. There has, as yet, been limited analysis of fisheye techniques giving empirical results. Fisheye views are still a recent research area, so the lack of formal analysis is not unexpected.

Furnas [6] originally evaluated the effect of a fisheye technique for navigation of a biological hierarchy. Twenty subjects were used and he presented the results in table 2.1. The results strongly

Viewing Technique	Percentage Correct
2 flat views	52
1 of each	64
2 fisheye views	75

Table 2.1: Results from Furnas'

suggest fisheye views are, or should be, a preferable visualisation technique in comparison to the flat view. Unfortunately the information type is hierarchically based, biasing visualisation toward an outline type viewer. This trial does indicate that hierarchical information can be searched, manually, most aptly with fisheye techniques. The distortion on the biological taxonomy gave an equivalent view to the distortion on the structured code in figure 2.2(b).

Keahey and Marley [14] compare a fisheye text viewer with a paging interface. Their viewer uses magnification of the text, reducing the font size as the distance from the focus increases, this is the same appearance based technique visually described in figure 2.2(a). Sixteen subjects were asked to initially read text with each interface, two texts were used, one was very structured: indentation, white space, and headings, the other was less structured: little white space and indentation, no headings. They were then queried regarding the content, firstly with no reference to the text allowed, and then with reference allowed. They produced the results in table 2.2, where the entries are the mean time and standard deviation. The results indicate that the fisheye

Article Type	Fisheye	Paging
unstructured	243.67 (154.62)	109.94 (45.40)
structured	70.94 (24.97)	120.71 (144.49)

Table 2.2: Results from Keahey and Marley

(magnifying) viewer is less useful than the paging viewer in navigation of unstructured text, while it is more useful in navigation of structured text. From a users perspective all subjects preferred reading the text in the paged view. Weir [29] utilised a similar fisheye text viewer and in his informal testing, noted that:

“Some users were observed to turn off the fisheye mechanism, ...”

indicating a problem with this form of distortion. As this study was an appearance based distortion method, repetition of this study, adding evaluation with the structural viewer in figure 2.2(b), would provide a reasonable validation for either technique.

Hollands *et al.* [12] represented a fictitious subway network, and evaluated a graphical fisheye technique against a scrolled “flat” view. Forty eight subjects were tested in this evaluation each participating in location tasks, optimal route tasks, and an optimal itinerary task. The results were mostly inconclusive as to the benefits of one technique over the other, though they suggest fisheye techniques are marginally better than scrolled techniques. This is a positive result for fisheye views as, at the least, fisheye views should be no worse than the conventional methods. The immaturity

of fisheye views at the time of testing, influences the results in favour of fisheye views. Sarkar and Brown [23] suggest their distortion method, based on the rubber sheet metaphor [25, 24], would improve the results in favour of the distortion method. Their rubber sheet metaphor, states the information can be stretched in different directions expanding/compressing the image size, similar to an image printed on an expanding balloon. The restriction, of course, is the screen dimensions stay fixed, so the push/pull effect occurs as mentioned in section 2.4.1.

Schaffer *et al.* [26] performed an analysis using similar techniques to Hollands. Twenty subjects were involved in a formal analysis, and a further two in an informal analysis. The subjects were confronted with a telephone network, and were asked to perform simple maintenance and navigation tasks. The results as reproduced in table 2.3, are much more conclusive in favour

View	Mean	Std. Dev.
<i>Running Time (seconds)</i>		
Fisheye	101.9	59.5
Full-zoom	161.2	71.6
<i>Number of Zooms</i>		
Fisheye	6.3	2.7
Full-zoom	10.9	4.3
<i>Successful completion of task as a ratio</i>		
Both	0.7	0.5

Table 2.3: Results from Schaffer *et al.*

of fisheye views. The time taken indicates the users are more efficient using fisheye views, and the lower number of zooms indicates navigation was easier in the fisheye view. Their approach was based on the hierarchical nature of the information, where they grouped, (clustered), nodes together creating an abstract “higher” node. This approach creates a form of folding view, where detail is enhanced by folding out the abstract nodes, or reduced by folding in the abstract nodes. A similar technique is implemented for many file browsers using direct manipulation, where a directory is folded/unfolded, typically, with a mouse action. Encapsulating information in this manner is a common top down or structural decomposition approach for problem solving, and facilitates extraction of outlines, (see section 2.4.4).

The results presented in this section, suggest fisheye views facilitate task oriented approaches, but may be less appropriate for sequential viewing, especially of text based documents.

2.4.3 Thresholds

Furnas proposed the idea of thresholds, where information points are either displayed or not, depending on their DOI value and the threshold value. Thresholding subdues information displaying the n most interesting points depending on their DOI values. The implication here is the DOI function assigns a continuous range of values, so n can be arbitrarily changed. Koike [16] discussed this issue and its consequences to the threshold technique. When assigning values to information points, via the DOI function, typically there is a symmetry to the assignment, where information points receive the same interest value. This equivalence of interest reduces the flexibility of thresholding, limiting the number of points that can be independently subdued. This limitation also creates large changes in the information as the threshold is adjusted. From a users perspective, large changes cause disorientation, as Mackinly *et al.*, [18] note:

“However, thresholding causes the visualisation to have gaps that might be confusing or difficult to repair. Furthermore, gaps can make it difficult to change the view. The desired destination from one view to another might be confusing as familiar parts of the visualisation suddenly disappear into gaps.”

The problem of removal and disorientation is related to the loss of global context that occurs when information is hidden. This is a trade off, not so much between detail and context, as

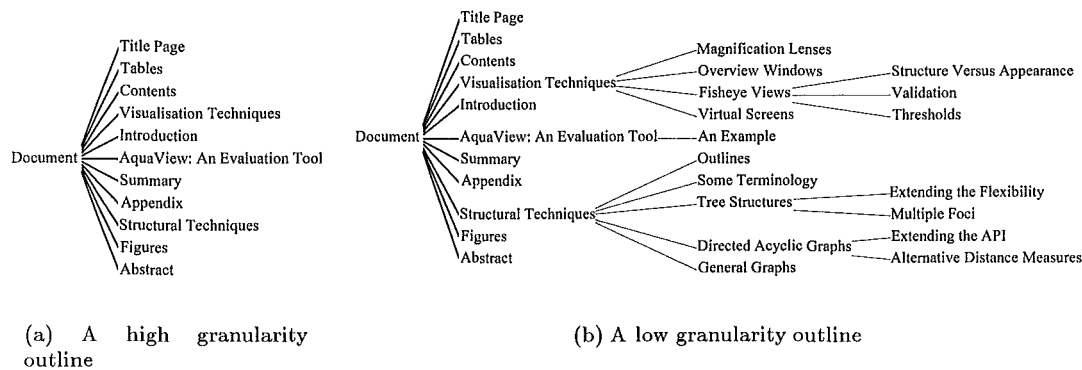


Figure 2.4: Possible outlines for an early version of this report

the typical scrolling/magnification/multiple window techniques do, but more between information overload and detail/context. The restriction to the screen space is a limiting factor that should always be kept in mind. Many distortion techniques try to map the entire information space onto the finite screen region. This approach is appropriate at times, but it can cause information overload, and overly limit the detail allowable.

Subduing information may cause disorientation, but this can be reduced by structural techniques using an appropriate outline. The outline needs to chunk the information together in a manner that reduces the large information shifts that occur in thresholding. The topic of outlines is now discussed in more detail.

2.4.4 Outlines

Extracting the outline of information is a crucial determinant in realising a useful structural distortion technique. In the dictionary [19] an outline is defined as:

“a general description of something which does not give all the details.”

Working with the outline of information is therefore, a reduction in the content. This reduction implies an encapsulation of information into broader categories. The categories can be linked together, to form a graph like structure containing nodes for the categories and edges for the connections.

Outlines are easy to generate, but that does not imply that they are adequate for the task. Figure 2.2(b) uses an outline based on the indentation of the statements. This indentation scheme, has inadequacies which become apparent as more complex information is used. The semantic details of the information are limited by this indentation approach, as the program increases in size, the number of child nodes at each level becomes large. This causes problems with the suppression of the information.

As mentioned in section 2.4.3, thresholding causes disorientation during visualisation. This problem can be alleviated by ensuring a valid description of the global context is retained. The granularity of the outline is related to the size of the information point. If an outline was extracted from this paper, a high granularity example is the chapters, (figure 2.4(a)) a lower granularity example is the chapters, sections and subsection, (figure 2.4(b)). These simple outlines retain a tree structure, where there is a single root and no cycles. This structural form is discussed in detail in section 4.2. Figure 2.4(b) can be thought of as an unfolded view of figure 2.4(a). For the high granularity example the children of each node is the text for the chapter. Typically there is a large quantity of text associated with each chapter, so there is too much detail to display on a single screen, let alone any context. The lower grained example reduces the text at each node, though it is still large. A better proposition may be to break the text for each section into paragraphs, reducing the text per node.

The choice of outline is a trade off between too much encapsulation and not enough. Retaining the semantic details of the information, requires some knowledge of the problem domain, and a little ingenuity.

Chapter 3

AquaView: An Evaluation Tool

Distortion techniques are best evaluated by a visual demonstration. To facilitate this visualisation a fisheye tool called **AquaView** was constructed. **AquaView** has a graphical user interface, (GUI), written using GraphScript [11] a tool for direct manipulation of graphs. Figure 3.1 shows the interface for **AquaView**.

GraphScript¹ is a Tcl/Tk [21, 30] extension for graph manipulation, simplifying node/edge manipulation and includes a number of graph theory, and layout algorithms. Tcl is an interpreted language and Tk includes a set of “widgets,” for constructing a GUI. Tcl/Tk facilitates rapid prototyping and is relatively simple to learn. GraphScript adds a subset of extra commands to the set of Tcl/Tk commands. This simplicity of implementation eased the fisheye extensions, and was considered the best package for the visual evaluation required.

An implementation criteria for fisheye views requires real time changes. While delay is unavoidable, more so in an interpreted language, Tcl/Tk allows for compiled code to be utilised. This feature was necessary for the implementation of the shortest path algorithms. An all pairs shortest path algorithm was required, the choice being Floyd’s [4], $O(n^3)$, algorithm, where n is the number of nodes in the graph. This algorithm is costly in both time and memory, an $O(n^2)$ adjacency matrix was required for computation and reference. Takaoka [28] mentions an $O(n^2 \log n)$ all pairs shortest path algorithm, which is presented in [27]. This approach will speed up evaluation of the shortest paths, but the memory requirements remain.

The time taken to load a 186 node tree, render it to the screen and evaluate the all pairs shortest path was approximately three minutes on a SPARCstation 4. For read only information this is a one time cost and the distortion times are in the order of seconds, it took about fifteen seconds to change the threshold view from the lowest to the highest, and less than five seconds to change the threshold from the highest to the lowest. When the information is being updated this delay is prohibitive, but as a demonstration tool it fulfils its purpose.

3.1 An Example

Robertson *et al.*, [22] presented a 3-dimensional visualisation technique, called cone trees for hierarchical information. They used the directory structure from a Unix file system as an example, which was found to contain approximately 600 directories and 10000 files. The tree formed was unbalanced and surprisingly flat, and would overwhelm a 2-dimensional display.

Directory structures are visually hierarchical, as an appearance based approach does not take advantage of this fact, a structural approach should be used. A hierarchical example will be used in the rest of this paper based on the sub-directory structure of a Unix file system. In fact the directory structure is a recursive listing of my home directory, containing 228 directories and 1264 files. **AquaView** is unable to handle this number of files, so a smaller subset is used. The reduced listing used can be found in appendix C.

¹GraphScript can be found on the web at: <http://www.fmi.uni-passau.de/Graphlet>

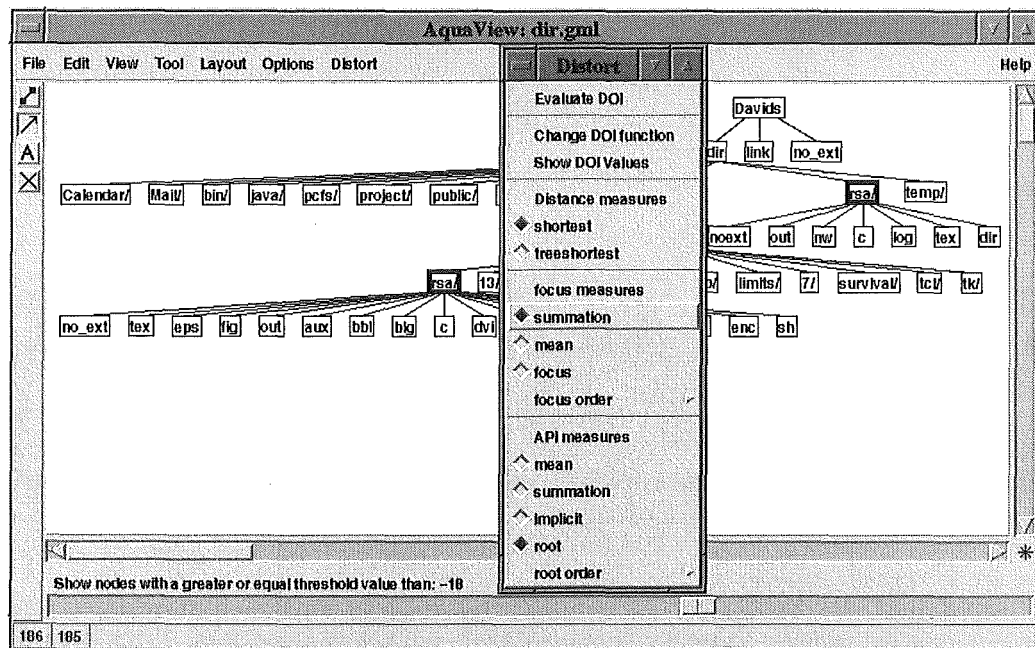
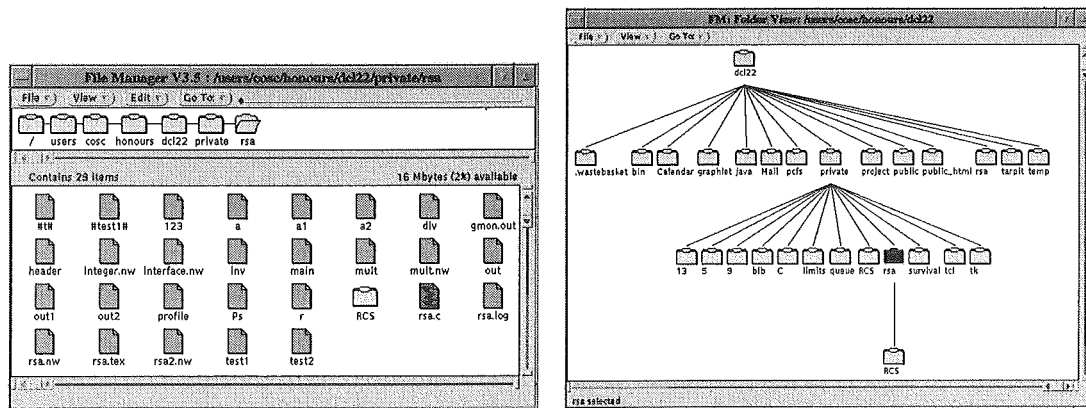


Figure 3.1: A screen shot of AquaView

An outline of the directory is required, and the basic outline as produced by `ls -R2` produces the parent child relationships required. A problem with this form of outline is the large width and relatively small depth, as mentioned by Robertson *et al.*, [22]. To reduce this problem, the encapsulation suggestion used by Schaffer *et al.*, [26] is incorporated. Files have attributes, and can be grouped accordingly. The grouping used for this example is by file type: directory; static link; or regular file. As the hierarchy consists of mainly regular files they are also grouped by extension. In an Unix environment the extension of a regular file is a major factor in determining its type. This grouping by extension is equivalent to a wild card, (regular expression), command line visualisation. The benefits of this grouping is the increase in depth and decrease in width it generally affords.

To demonstrate the necessity for a detail and context directory browser, figure 3.2 shows a multiple window solution that has a single context browser and detail window. The upper part of figure 3.2(a)'s screen displays a zero order fisheye view, (see section 4.2) which is the context, the bottom screen displays the details of the current directory. Figure 3.2(b) is an overview viewer displaying a fuller context view, which is currently a first order fisheye view of the directories. The physical screen size necessary to display the overview view, quickly expands to an unviewable size. Visualisation of multiple directories simultaneously, requires an additional detail viewer, consuming even more screen space, creating overlapping windows and causing information overload. In section 4.2 a fisheye view of two simultaneous directories will be displayed retaining both context and detail while consuming less space than the overview viewer.

²The Unix command `ls -R` produces a recursive listing of files and subdirectories beginning at some named directory.



(a) The detail view

(b) The overview view

Figure 3.2: Screen shots of filemgr

Chapter 4

Structural Techniques

Information is structured, whether visibly apparent or some underlying interconnection. This structure can be used for the distortion, allowing a greater semantic view. When pictures taken with a fisheye lens are viewed the structure is visible, where the size and physical proximity of objects in the flat view imply the size and proximity in the distorted view.

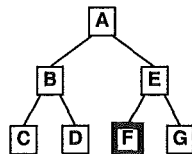
In this chapter we look at distortion based on the underlying structure of the information. Let $G = (V, E)$ be a *graph* where $V = \{v_1, v_2, \dots, v_n\}$ is the set of vertices, (or nodes), and $E \subseteq V \times V$ is the set of edges. The underlying structure that we discuss will be represented by some graph G , as extracted from the information.

4.1 Some Terminology

In discussing the models some terminology is required for convenience. The terminology that will be used is:

- \mathbf{f} will be used to denote the focus point, where there are multiple foci, $\mathbf{f} = \{f_1, f_2, \dots, f_n\}$, will be a set, otherwise it is a singleton.
- \mathbf{r} will be used to denote the root, when there are multiple roots, $\mathbf{r} = \{r_1, r_2, \dots, r_n\}$, will be a set otherwise, it is a singleton.
- $\mathbf{DOI}(\mathbf{x}|\mathbf{f})$ is the degree of interest for node x given the focus is f .
- $\mathbf{API}(\mathbf{x})$ is the a priori interest of node x .
- $\mathbf{D}(\mathbf{x}, \mathbf{y})$ is the distance between nodes x and y .
- $\mathbf{PN}(\mathbf{x}, \mathbf{y})$ is the set of all nodes that are in the shortest path between x and the y including x and y . $\mathbf{PN}(\mathbf{x}, \mathbf{r})$ will commonly be referred to as root path nodes for x .
- Two nodes x and y are **semi-equivalent** if $D(p_1, x) = D(p_2, y)$, where $D(p_1, x) = \min\{D(p_i, x)\}$, and $D(p_2, y) = \min\{D(p_i, y)\}$, $\forall p_i \in \mathbf{PN}(y, \mathbf{r})$. $y \equiv \mathbf{f}_{\text{focus}}$
- Two nodes x and y are **equivalent** if they are semi-equivalent and $D(x, \mathbf{r}) = D(y, \mathbf{r})$.

Example 1 In the following figure $\mathbf{f} = F$, $\mathbf{PN}(A, F) = \{A, E, F\}$, nodes B and G are semi-equivalent, and nodes C and D are equivalent.



4.2 Tree Structures

The first structural form for distortion is based on a tree structure. A tree can be considered as a digraph, where the edges are ordered pairs, indicating the parent child relationship. The restriction for trees are:

- R1. $\exists(r, v)$ where $v \in V$, (a root),
- R2. $\forall v_j \in V - r, \exists(v_i, v_j) \in E$ where $v_i \in V$, (fully connected),
- R3. \forall paths $v_i \rightarrow w_1 \rightarrow w_2 \rightarrow \dots \rightarrow w_n \rightarrow v_j$, and $v_i \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_m \rightarrow v_j$, $u = m$ and $w_1 = u_1, w_2 = u_2, \dots w_n = u_m$, (single path between any two nodes), and
- R4. \nexists paths $v \rightarrow w_1 \rightarrow w_2 \rightarrow \dots \rightarrow w_n \rightarrow v$ (no cycles).

Furnas [6] proposed the following DOI constructs for tree structures. The distance between two nodes x and y is the number of edges in the shortest path between x and y . Alternatively:

$$D(x, y) = |PN(x, y)| - 1$$

The a priori interest of node x is related to its distance from the root of the tree, r , and is defined as:

$$API(x) = -D(x, r)$$

Therefore the degree of interest for node x is defined as:

$$DOI(x|f) = -D(x, r) - D(x, f) \quad (4.1)$$

having the following properties:

1. semi-equivalent nodes have the same DOI value, (hence, so do equivalent nodes),
2. $PN(f, r)$, the root path nodes, have the highest DOI value, and
3. as the distance from the root path nodes increases the DOI values decrease.

Properties 2 and 3 imply an ordering for the DOI function. The threshold level can be defined in terms of the ordering — a zero-order view is the distorted view corresponding to the highest threshold. In general the i^{th} -order view corresponds to the view given by the $(i+1)^{th}$ highest threshold.

Furnas [6] suggested evaluation of the DOI function for a shift in focus for y to y' was reduced computationally:

“...since the whole DOI function above¹ their common ancestor is unchanged.”

This statement is false, it can be shown that when the shift of focus from y to y' is made and the common ancestor is A , then the DOI values for nodes above the common ancestor are unchanged, if and only if $D(y, A) = D(y', A)$.

It has been pointed out that equation 4.1 is inflexible [16]. In section 4.2.1 these problems are identified and addressed.

4.2.1 Extending the flexibility

There are a number of inflexibilities in equation 4.1 that need to be addressed.

1. By not differentiating between equivalent and semi-equivalent nodes, there is a lack of distinct DOI values. This reduces the number of nodes that can be subdued, hence requiring more nodes to be displayed than is necessary, (a case of information overload).

¹A node x is above a node y if y is not an ancestor of x .

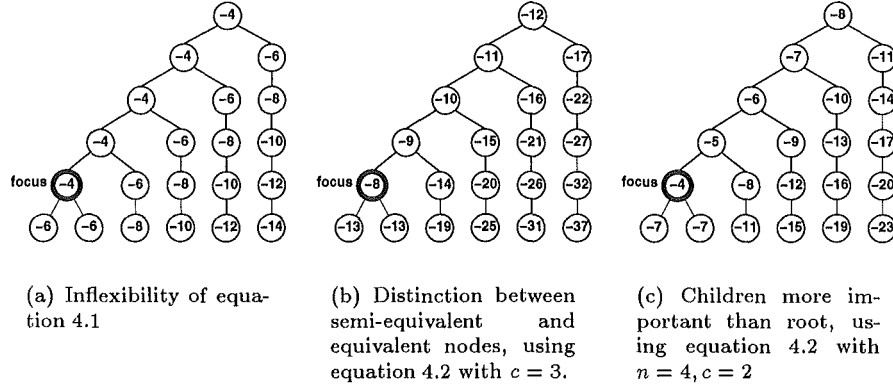


Figure 4.1: Visual comparison of functional forms for tree structures

2. The root of the tree is always more important than the children of the focus, irrespective of the distance between the root and the focus. This problem is associated with the distinction between equivalent and semi-equivalent nodes.
3. When visualising one part of the information, another section is important that may be in a different branch of the tree. Rather than require the threshold to be high enough to display both portions — inducing information overload — multiple foci need to be supported.

The following equation addresses inflexibilities 1 and 2.

$$DOI(x|f) = -(c-1)D(x, r) - cD(x, f), c > \frac{1}{2} \quad (4.2)$$

and retains the ordinal properties 2 and 3 in section 4.2, (see appendix A for proof). Property 1 in section 4.2 is no longer true for equation 4.2, and can be replaced with:

1. semi-equivalent nodes have different DOI values, (unless they are also equivalent).

The constant c determines the ordinality properties, and in appendix A the following is derived:

- if $c > \frac{D(r,f)+1}{2}$, the children of the focus have a smaller DOI value than the root path nodes.
- if $c < \frac{n+1}{2}$, the root path node that is a distance of n , from the focus, has a smaller DOI value than the children of the focus.

In figure 4.1 three different sets of DOI values are displayed for the same tree structure. Figure 4.1(a) shows the lack of uniqueness in the DOI values. Reducing the threshold in this case results in large shifts in the display. In comparison, figures 4.1(b) and 4.1(c) differentiate between the semi-equivalent and equivalent nodes, reducing the disorienting effect of a threshold change. Figure 4.1(c) also displays the flexibility which allows children of the focus to have an increased importance.

Koike [16] presented a technique based on fractals, that kept the number of nodes displayed, *nearly* constant. This approach is based on the distance measure only, and does not use an *a priori* importance measure. As equivalent nodes have the same value for equations 4.1 and 4.2, a node which has more children will consume more space than one with less, Koike identified this as a *sibling overload* problem. Koike's method based the importance of a node on the number of siblings and its parents importance. The greater the number of siblings the less important the node is. This approach allowed for an increased flexibility in the number of threshold values, but it is not as flexible as equation 4.2, which takes into consideration the *a priori* interest of the nodes.

The final inflexibility, relating to multiple foci is a more difficult problem. In section 4.2.2 this problem is addressed.

4.2.2 Multiple Foci

Many interfaces allow for multiple windowing or split screens, mainly for the concurrent interaction of different parts of the information. This form of interaction is typically insufficient for the purpose, where the user requires the detail of the multiple information segments and the relationship(s) between them. Interacting with the information in this way, can be supported with multiple foci.

Group aware applications are a major research area, the increasing bandwidth of communication links and accessibility of co-participants makes for concurrent interaction over long distances feasible. When concurrent interaction occurs, location awareness of participants is typically required. Multiple foci help alleviate this problem, retaining detail and context while displaying participants location, by representing each participant by a focus. Greenberg *et al.*, [9] discusses this workspace awareness problem, comparing three techniques: an offset lens; a head up lens; and a fisheye lens. The fisheye lens system is a structural distortion technique for text, similar to the systems used by Keahey [14] and Weir [29], figure 2.2(a) on page 10 was created using this system. The addition of multiple foci allows a participants location to be determined by the increased size of the text near their focus. The detail shown for the alternate participants was user controllable, though location awareness was lost when the text was scrolled. This problem, again, emphasises the problems with an appearance based technique for text visualisation, although this problem would also occur if a participants location was in a subdued region in a structural technique.

Multiple foci requires changes to the distance measure component of equations 4.1 and 4.2 to accommodate the focus vector. For equation 4.2 the constant, c , also needs changing to accommodate the focus vector. Mitta and Gunning [20] proposed changing the distance measure to a summation of distances from x , to each of the foci. That is:

$$D(x, \mathbf{f}) = \sum_{i=1}^n D(x, f_i).$$

This is the best approach, although there are problems with the ordering. This approach has the following properties:

- the foci no longer have the maximum DOI values,
- a *center* node² exists which has the smallest DOI value, and
- nodes not in the direct path between any two foci, or a foci and the root, may have larger DOI values than them.

This implies that displaying the foci requires more nodes to be viewed. This is the reduction in flexibility that has been previously discussed, and can cause information overload.

A Statistical Approach

In Bayesian [7] statistics the concepts of *posteriori* and *a priori* distributions are essential for statistical inference. Given the *a priori* distribution and the world state, the *posteriori* distribution can be evaluated. The initial estimate of the *a priori* distribution is subjectively chosen, thereafter the previous *posteriori* distribution can be used.

This technique allows for a narrowing of the *a priori* distribution, a learning based on the previous information and the current state of nature. It is an interesting analogy that can be utilised in the evaluation of the DOI function. If the DOI is considered as the *posteriori* function, after the initial *a priori* estimate is made, the DOI values can be used as the next *a priori* estimate. The main motivation for this form of “learning” is based on the assumption that the new focus is based on the previous focus. When navigating information it is unusual to stop at random points, the likelihood is that the current focus induces the next reference point(s).

²The center node is the node that has the smallest distance measure.

Figure	Minimum	Maximum	Range	Threshold value used
4.2(a)	-22	-5	17	-11
4.2(b)	-36	-9	27	-18
4.2(c)	-30	-6	24	-16
4.2(d)	-94	-38	56	-16

Table 4.1: Thresholds used in figure 4.2

Churcher [3] described a multiple focus technique that is similar in format. His system **Photi**, which is a graphical distortion system using the approach by Sarkar and Brown, transforms a possibly already distorted view, regarding it as effectively flat. This technique, he proposes, “side-steps” the false foci that can occur with multiple focuses.

Figure 4.2(c) shows a hierarchical view of a directory structure, using this approach for two foci. This approach is equivalent to utilising a DOI function of:

$$DOI(x|\mathbf{f}) = DOI(x|f_1, f_2, \dots, f_n) = -nD(x, r) - \sum_{i=1}^n D(x, f_i) \quad (4.3)$$

corresponding to equation 4.1 or

$$DOI(x|\mathbf{f}) = DOI(x|f_1, f_2, \dots, f_n) = -n(c-1)D(x, r) - \sum_{i=1}^n cD(x, f_i) \quad (4.4)$$

for equation 4.2. For these equations f_1 is the current focus and f_2, f_3, \dots, f_n are the $n-1$ previous foci. This approach takes into account all of the preceding foci up to and including the current focus. The presumption was that the last focus of interest is an influence in determining the current focus point. This property is similar to the memoryless property of a Markovian process, where the current state is dependent on the last state only. This approach suggests that equation 4.3 become:

$$DOI(x|f_1, f_2, \dots, f_n) = DOI(x|f_1, f_2) = -2D(x, r) - D(x, f_1) - D(x, f_2) \quad (4.5)$$

and equation 4.4 become:

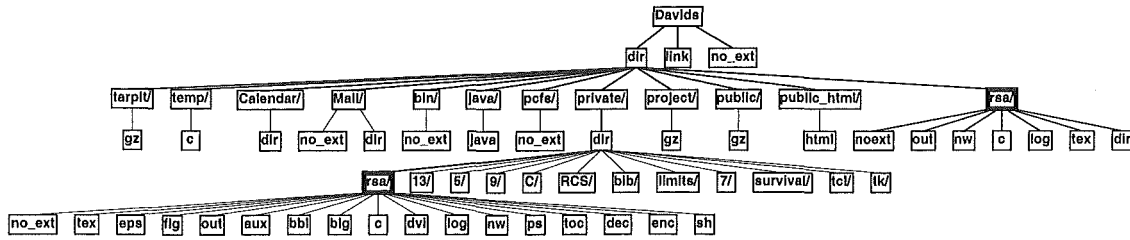
$$DOI(x|f_1, f_2, \dots, f_n) = DOI(x|f_1, f_2) = -2(c-1)D(x, r) - cD(x, f_1) - cD(x, f_2) \quad (4.6)$$

When $c = 2$ equation 4.6 is equivalent to equation 4.1.

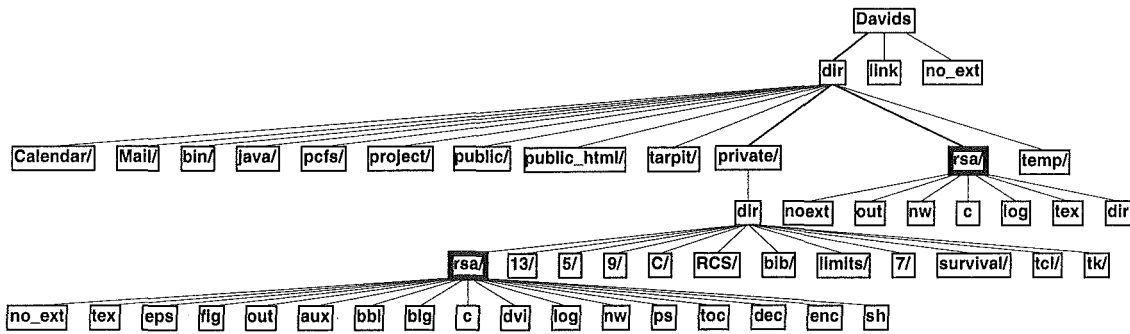
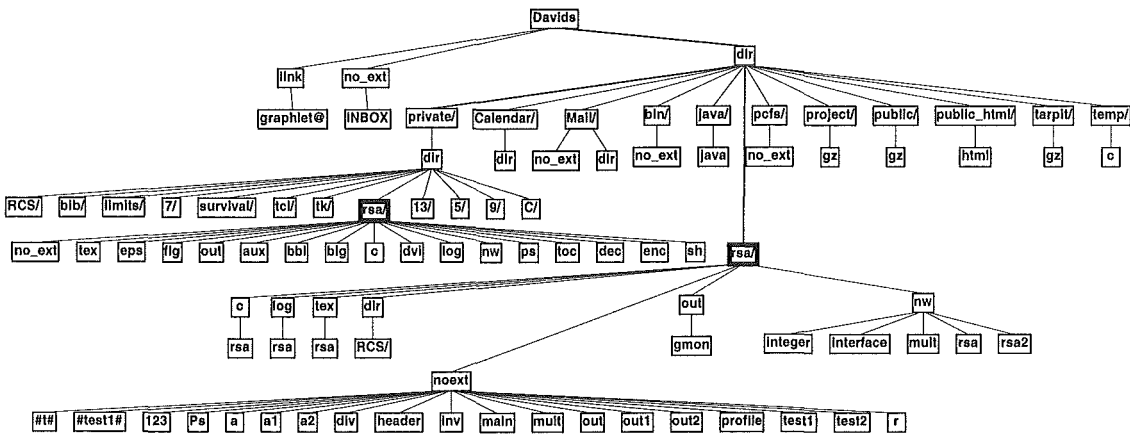
Example Views

Returning to the example of directory browsing, figures 4.2(a), 4.2(b), 4.2(c) and 4.2(d) are different views of a directory hierarchy with multiple foci. This example is equivalent to visually comparing the contents of two directories with the same name — in this example the directory being compared is named **rsa**. The emboldened border width identifies the two foci. In comparison to the view displayed by **filemgr**, context and detail are retained while limiting the displayed information. The interface for this information is inelegant and a better presentation would make a better illustration, but the effect is sufficient as a demonstration. Figure 4.2(a) displays information at a deeper level than is required. This increased information can overwhelm the display as figure 4.2(c) does. As was identified previously, the lack of threshold values is a contributing factor for the information excess problem. Table 4.1 displays the range of threshold values and the selected threshold for each of the figures. Figure 4.2(b) displays a pruned tree showing great detail and context, without an overwhelming amount of information. The full view of figures 4.2(a) and 4.2(b) were able to be displayed consuming less than half of the screen space³. Figure 4.2(c)

³The display was a Sun Microsystems color monitor NCE Model 447L, with a screen size of 325 × 254mm.



(a) Furnas' approach, (DOI function 4.1)

(b) Extended approach, (DOI function 4.2, $c = 2$)

(c) Bayesian approach, (DOI function 4.5)

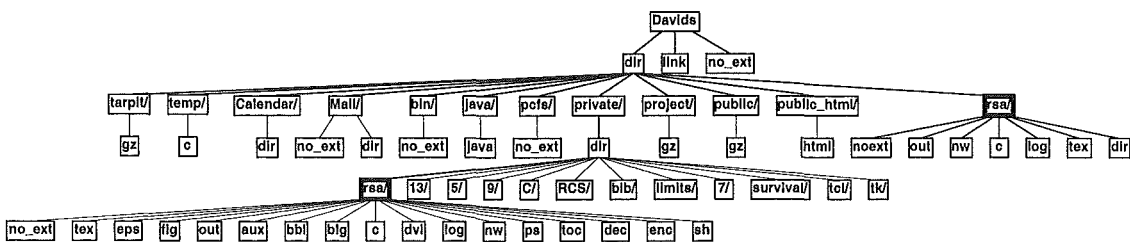
(d) Bayesian approach, (DOI function 4.6, $C = 4$)

Figure 4.2: Multiple foci views

was unable to be displayed in its entirety, as the width exceeded the available window size, hence the artificial levelling effect for the view. The corresponding `filemgr` view required two file windows and the overview window, which also exceeded the screen dimensions. The advantage of grouping by file type and extension is unavailable for `filemgr`, so the views are not equivalent, but this demonstrates the effectiveness of a semantic view, and especially the outline capabilities. An interesting point is the equivalence of the views for figures 4.2(b) and 4.2(d). Figure 4.2(d) has a greater range in threshold values, and is therefore more flexible, yet the semantic details have been reduced, as shown by the excess irrelevant detail shown.

4.3 Directed Acyclic Graphs

Many structures are based on trees, but most are not. In this section the structural form is extended to *directed acyclic graphs*, (DAGs). Directed graphs which are acyclic do not satisfy restriction R3 on page 19. The removal of restriction R3 induces *diamonds*, (see figure 4.3), and changes to the distance measure, which is discussed more fully in section 4.3.2 are required. DAGs also affect property R1 on page 19 allowing multiple roots. Allowing multiple roots implies the a priori measure, as it has been defined, needs updating, in section 4.3.1 the merits of different adjustments are discussed.

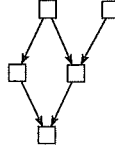


Figure 4.3: A DAG with two roots and a diamond

4.3.1 Extending the API

When a tree is extended to allow for multiple roots, the API function needs to be redefined to accommodate this change. Furnas and Zacks [5] discuss multitrees, which are acyclic directed graphs, with the limitation that no *diamonds* are allowed. A diamond occurs in a graph whenever there are two distinct paths between two nodes. Each node, n , in a multitree has the property that the descendants of n form a tree with n as the root — such a tree will be referred to as n 's descendent tree.

This limitation allows for equations 4.1 and 4.2 to be used with the single change to the API function. The solutions for changes to the API are:

1. the negative mean distance from the roots: $API(x|f) = -D(x, \mathbf{r}) = -\frac{\sum_{i=1}^n D(x, r_i)}{n}$,
2. the negative sum of all the distances from the roots: $API(x|f) = -\sum_{i=1}^n D(x, r_i)$,
3. the distance to the i^{th} closest root from the focus: $API(x|f) = -D(x, r_i)$, and
4. the distance from an implicit root: $API(x|f) = -D(x, r_I)$.

Figures 4.4(a)–4.4(l) display the DOI values for a digraph — in fact multitrees — using the different API measures. Table 4.2 summarises the properties of the functional forms used. The description for the properties evaluated is:

Focus most important: is true if the focal point has the highest degree of interest.

Root paths greatest: is true if all the nodes in the paths to the implicit root can have higher interest values than nodes not in these paths.

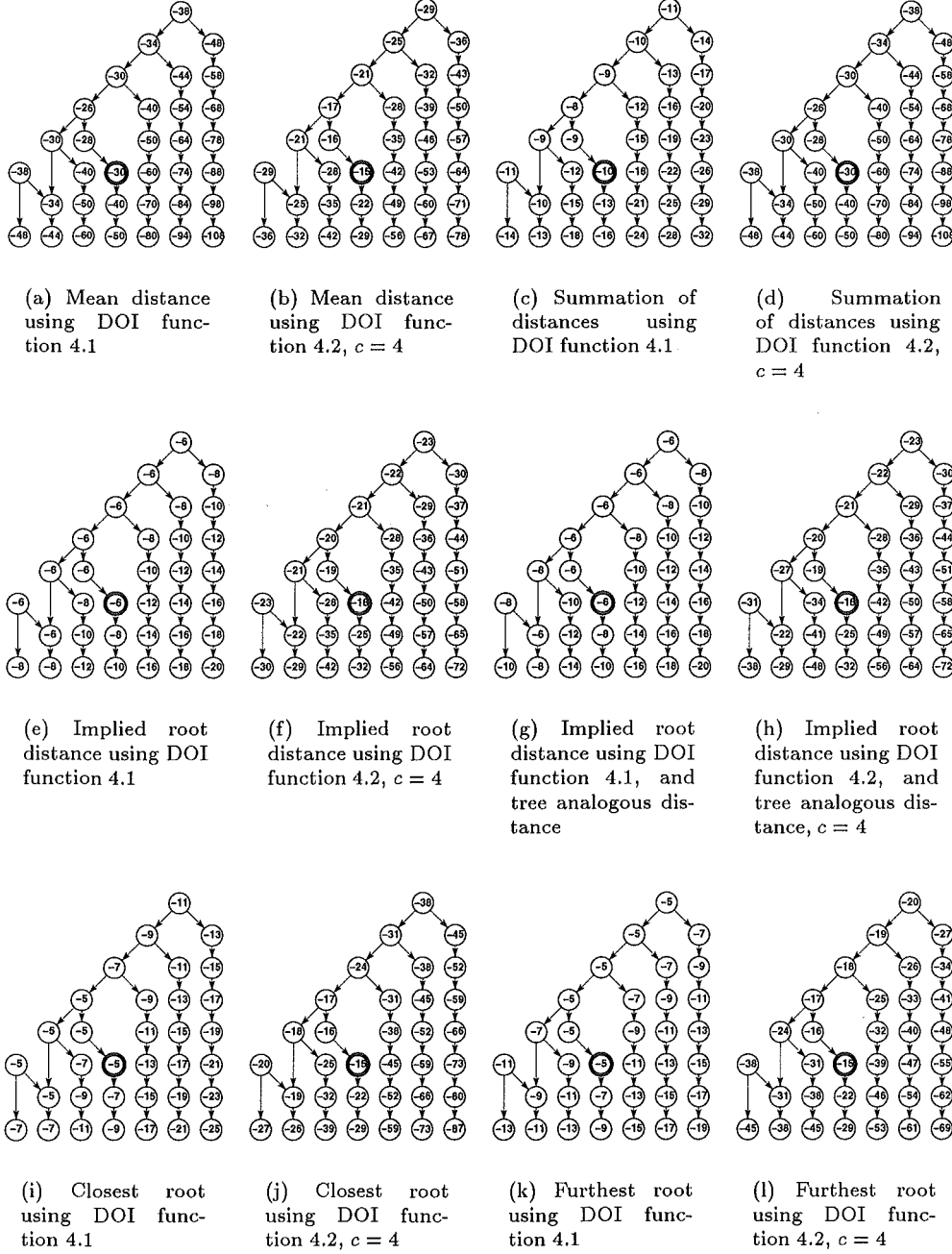


Figure 4.4: Comparison of distance measures for multiple roots.

Property	Mean Distance	Summation	Implied Root	Closest	Furthest
			Shortest Tree		
Focus most important	yes	no	yes	yes	yes
Root path greatest	no	yes	yes	no	no
Flexible	yes	yes	yes	yes	yes
Extensible child interest	no	yes	yes	yes	yes

Table 4.2: Comparison of functional forms for multiple roots

Flexible: is true if the interest values are different for semi-equivalent nodes, using equation 4.2.

Extensible child interest: is true if the interest of the child can be increased to a value above the root path nodes, (note: this feature does not imply the second property is violated).

The best solution is the implied root method using the shortest path distance, from the table. An implicit root is an artificial node which is the parent of the explicit roots. The outline extracted in section 2.4.4, based on this report structure made use of an artificial node called Document. Examples of this form of single rooted digraph is an object oriented class lattice, or a task schedule. In table 4.2, this method has two sub-headings, *Shortest* and *Tree*, which are associated with the distance measure used for the API values. As there are multiple paths between nodes, due to the introduction of diamonds, (see figure 4.3), the choice of distance measure between nodes, was not obvious. A discussion is now made on alternative distance measures.

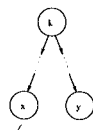
4.3.2 Alternative Distance Measures

For trees there is only a single distance measure possible, based on the paths between nodes, excluding repeated links. For DAGs this measure becomes more difficult to define. Paths in trees have the following form:

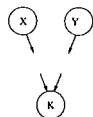
- a path can go from x directly down to y , or



- from x up to k and then from k down to y , but not



- from x down to k and then from k up to y .



In figure 4.2 an analogous measure to this was utilised for the implicit root method. The results suggest this method is inadequate for this simple tree extension, as nodes not in the explicit root paths may have a higher interest.

To illustrate the utility of the tree measure, figure 4.5 compares a directory with a link as the focus. The link is connected to the file it references, in this case it is a shortcut to a lower directory. Figure 4.5(c) is a view using the analogous tree measure. In this case the “real” parent

of the directory is subdued, as opposed to figure 4.5(a) where it is displayed. As the threshold is reduced from figure 4.5(a) to figure 4.5(b) a gap appears, giving a disorienting affect described by Mackinly *et al.*, [18], (see section 2.4.3). These two approaches are complimentary rather than contradicting and the measure to use depends on the situation. If, for example, the “real” parent of the linked file was required, the shortest path measures is more appropriate as the detail required is less than an order-1 view, whereas in the tree analogous measure the detail required is exorbitant. In contrast if the linked file itself is only required, typical if the link is a directory, the shortest path displays too much information — though this excess can be subdued with an appropriate threshold selection.

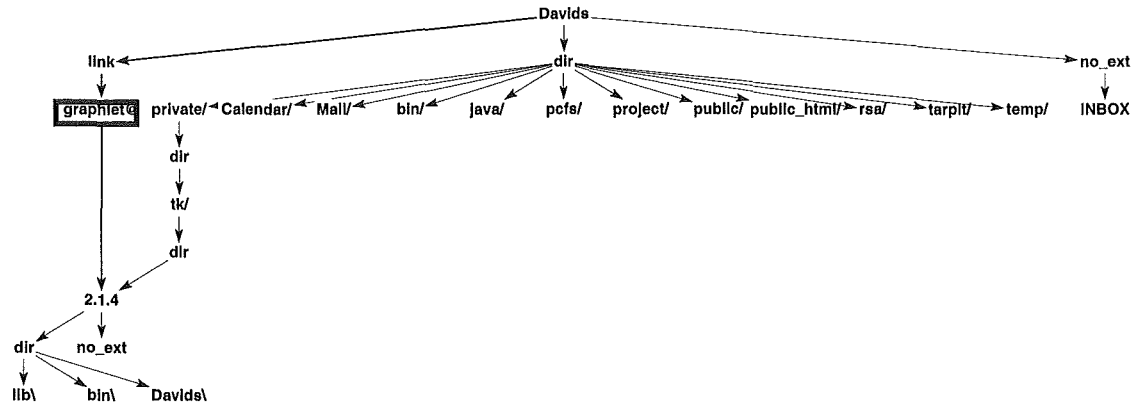
In appendix B an algorithm for the tree analogous all shortest paths is presented.

4.4 Cyclic Digraphs and General Graphs

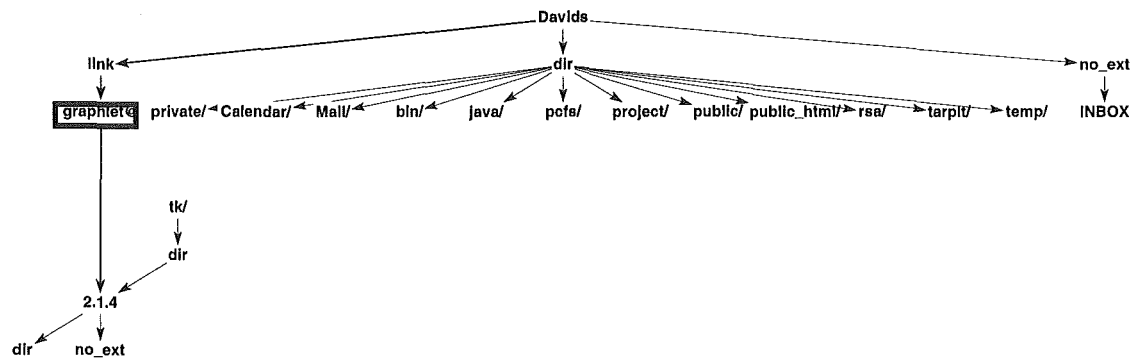
Removing the acyclic limitations of DAGs, allows for general graphs as a structural representation. Allowing cycles in a structure presents an overwhelming problem for structural distortion techniques. The semantic description presumed by the structure is incoherent to the distortion techniques utilised in sections 4.2 and 4.3. This problem is associated with the outline constructed and may best be reduced by the technique described by Schaffer *et al.*, [26], where nodes are chunked together, creating a folding layer view. It seems likely that such graphs are best suited to appearance based methods, as the link measures have lost their semantic appeal.

When the graph is *nearly acyclic* — the number of cycles are few — a structural distortion is still possible. The cycles in a nearly acyclic graph can be distorted with the same techniques utilised for acyclic graphs. It is possible that there is no root, due to the nature of the cycle. This loss of an identifiable root may best be solved using Furnas *et al.*, [5] solution, for multitrees where selection of a focal content and context node is required by the user. As the root distance is just the basis for the API measure, this subjective user action is appropriate. The problem arises when large quantities of information is displayed so the appropriate choices can be made.

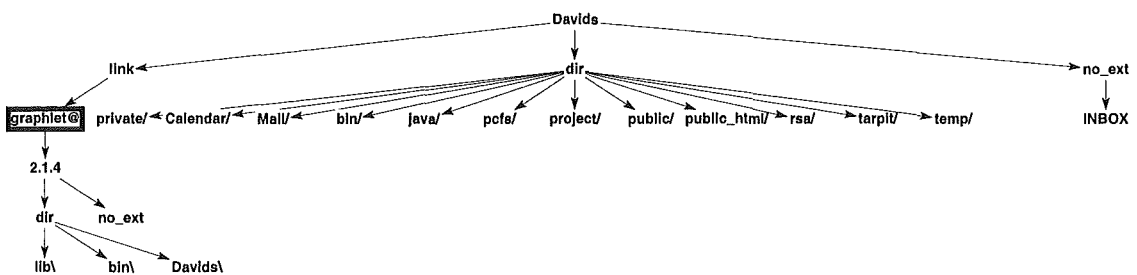
An example of a cycle is shown in figure 4.4, using the directory data, where the focus is again a link, this time referencing the root. Selecting the focus as the context, is equivalent to multiple foci without an API measure. This limits the semantic representation, and is an argument against structural views for cyclic, and hence general graphs.



(a) Distortion using shortest distances, threshold = -12



(b) Distortion using shortest distances, threshold = -10



(c) Distortion using tree analogous distances, threshold = -12

Figure 4.5: Distance methods for acyclic digraphs

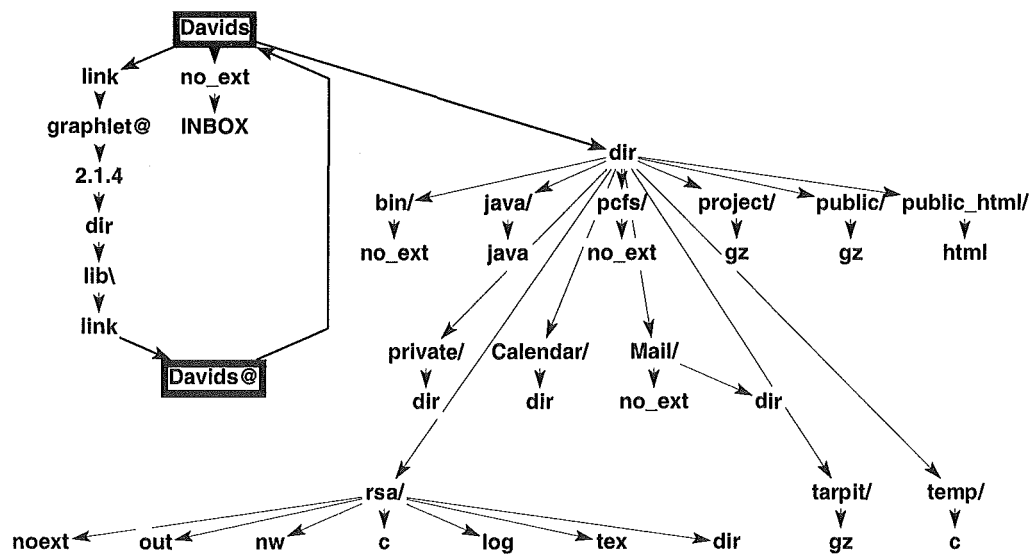


Figure 4.6: A cyclic digraph

Chapter 5

Conclusion

5.1 Future Work

The effectiveness of fisheye techniques is still a matter of contention. The feeling is they will add new dimensions to information visualisation and creation that can not be matched by non-detail and context techniques. Further analysis of these techniques should be made in a thorough and statistically sound manner.

The fisheye components for a file manager are available from this paper. Implementation of such a system should be relatively straight forward, and with the abundance of direct manipulation file managers around, a comparative study on the benefits of either technique should be next carried out.

Structural techniques for supporting general graphs, have been shown to be inappropriate with the expansions and suggestion made, but this is still an open question as to whether the structure can be used, perhaps the structure of the structure?

The problems identified with the lack of distinction between semi-equivalent and equivalent nodes in section 4.2, needs to be extended to differentiation between equivalent nodes. The method suggested in this paper: encapsulation via the outlining tool, may not be sufficient.

5.2 Summary

Fisheye techniques have a widespread application domain, whenever you are confronted with large amounts of information, either textual or graphical, consider how your cognitive processes are reduced because of the lack of integration of context and detail. A structural distortion should enable the semantics of the information to be more readily accessible, helping visualisation. This paper has proposed a number of extensions to the original ideas forwarded by Furnas, especially for tree-like structures and will allow flexible distortions. This flexibility is the allowance of context and detail while reducing irrelevant information. Outlining methods, need to be restricted to producing structural forms which are “nearly” acyclic. General graphs are too complex for the structural techniques presented in this paper.

Bibliography

- [1] M Behzad and G Chartrand. *Introduction to Theory of Graphs*. Allyn and Bacon Inc., Boston, 1971.
- [2] E. A. Bier, M. C. Stone, K Fishkin, W Buxton, and T Baudel. A taxonomy of see-through tools. In *Proceedings of CHI'94 Conference on Human Factors in Computing Systems* Boston, April 24–28, pages 358–364. Addison-Wesley, 1994.
- [3] Neville Churcher. Photi—a fisheye view of bubbles. *Information and Software Technology*, 37(1):31–37, 1995.
- [4] R Evans and E Minieka. *Optimizing Algorithms for Networks and Graphs*. Marcel Dekkar, Inc., 1992.
- [5] G. W. Furnas and J Zacks. Multitrees: Enriching and reusing hierarchical structure. In *Proceedings of CHI'94 Conference on Human Factors in Computing Systems* Boston, April 24–28, pages 330–336. Addison-Wesley, 1994.
- [6] GW Furnas. Generalized fisheye views. In *Human Factors in Computing Systems III. Proceedings of the CHI'86 conference.*, pages 16–23. Amsterdam; North Holland/ACM, 1986.
- [7] P Gärdenfors and N Sahlin. *Decision Probability and Utility*. Cambridge University Press, 1988.
- [8] R Gould. *Graph Theory*. The Benjamin/Cummings Publishing Company, Inc., 1988.
- [9] S Greenberg, C Gutwin, and A Cockburn. Using distortion-oriented displays to support workspace awareness. Technical Report 96/581/01, Department of Computer Science, University of Calgary, Calgary, CANADA., 1996.
- [10] D Halliday and R Resnick. *Fundamentals of Physics*. John Wiley & Sons, Inc, 1988.
- [11] M Himsolt. *The GraphScript Language*, 1996.
- [12] J. G. Hollands, T. T. Carey, M. L. Matthews, and C. A. McCann. *Presenting a graphical network: a comparison of performance using fisheye and scrolling views*. Elsevier, Amsterdam, 1989.
- [13] T. A. Keahey and E. L. Roberston. Non-linear image magnification. Technical Report 460, Department of Computer Science, Indiana University, 1996.
- [14] TA Keahey and J Marley. Viewing text with non-linear magnification: An experimental study. Technical report, Department of Computer Science, Indiana University, April 1996.
- [15] B Kernighan and D Ritchie. *The C Programming Language*. Prentice Hall, 1988.
- [16] H Koike. Fractal views: A fractal-based method for controlling information display. *ACM Transactions on Information Systems*, 13(3):305–323, July 1995.

- [17] YK Leung and M Apperley. A review and taxonomy of distortion-oriented presentation techniques. *ACM Transactions on Computer Human Interaction*, 1(2):126–160, 1994.
- [18] J. D. Mackinlay, G. G. Robertson, and S. K. Card. *The perspective wall: detail and context smoothly integrated*. ACM, 1991.
- [19] WT McLeod, editor. *The Collins Dictionary and Thesaurus*. William Collins Sons & Co Ltd, 1987.
- [20] D Mitta and D Gunning. Simplifying graphics-based data: applying the fisheye lens viewing strategy. *Behaviour & Information Technology*, 12(1):1–16, 1993.
- [21] JK Ousterhout. *An Introduction to Tcl and Tk*. Addison-Wesley, 1993.
- [22] G. G. Robertson, J. D. Mackinlay, and S. K. Card. *Cone Trees: Animated 3D visualisations of hierarchical information*. ACM, 1991.
- [23] M Sarkar and MH Brown. Graphical fisheye views of graphs. In *Proceedings of CHI'92 Conference on Human Factors in Computing Systems* Monterey, May 3–7, pages 83–91. Addison-Wesley, 1992.
- [24] M Sarkar and S. P. Reiss. Manipulating screen space with stretch tools: Visualising large structure on small screen. Technical Report CS-92-42, Department of Computer Science, Brown University, Providence, USA, 1992.
- [25] M Sarkar, S. S. Snibbe, O. J. Tversky, and S. P. Reiss. Stretching the rubber sheet: A metaphor for viewing large layouts on small screen. Technical report, Department of Computer Science, Brown University, Providence, USA, 1992.
- [26] D Schaffer, Z Zuo, S Greenberg, L Bartram, J Dill, S Dubs, and M Roseman. Navigating hierarchically clustered networks through fisheye and full-zoom methods. *ACM Transactions on Computer Human Interaction*, 3(2):162–188, 1996.
- [27] T Takaoka. An all pairs shortest path algorithm with expected time $o(n^2 \log n)$. *SIAM Jour. Comp.*, 16(6):1023–1031, 1992.
- [28] T Takaoka. Shortest path algorithms for nearly directed graphs. Technical Report TR-COSC 02/97, University of Canterbury, 1997.
- [29] P Weir. Distortion oriented workspace awareness, 1996.
- [30] B.B. Welch. *Practical Programming in Tcl and Tk*. Prentice-Hall PTR, 1985.
- [31] R Wilson. *Introduction to Graph Theory*. Bell & Bain Ltd., Glasgow, 1972.
- [32] I Witten, A Moffat, and T Bell. *Managing gigabytes : compressing and indexing documents and images*. Van Nostrand Reinhold, 1994.

Appendix A

Proving the Ordinality of Equation 4.2

A.1 Ensuring Focus has Highest DOI

Given focus f and point $x \neq f$ then it needs to be shown that $DOI(f|f) > DOI(x|f)$.

$$\begin{aligned}
DOI(f|f) &> DOI(x|f) \\
(c-1)API(f) &> (c-1)API(x) - cD(x, f) \\
-(c-1)D(f, r) &> -(c-1)D(f, x) - cD(x, f) \\
-cD(f, r) + cD(x, r) + cD(x, r) &> D(x, r) - D(f, r) \\
c[D(x, r) - D(f, r) + D(x, f)] &> D(x, r) - D(f, r) \\
c[D(x, p) + D(p, r) - D(f, p) \\
-D(p, r) + D(x, p) + D(p, f)] &> D(x, p) + D(p, r) - D(f, p) - D(p, r) \\
c &> \frac{D(x, p) - D(f, p)}{2D(x, p)} \\
c &> \frac{1}{2}
\end{aligned}$$

Therefore, if $c > \frac{1}{2}$ then $DOI(f|f) > DOI(x|f), \forall x \neq f$. \square

A.2 Ensuring Root Path Nodes have Second Highest DOI

To show that the root path nodes, $PN(x, r)$ have the second highest DOI values we state the following lemma:

Lemma 1 If c_f is a child of the focus and $c \geq 1$ then $DOI(c_f|f) > DOI(x|f), \forall x \notin PN(f, r)$, where x is not also a child of the focus.

Proof

Let $x \notin PN(f, r)$ and c_f be a child of the focus f . Let $P \in PN(f, r) - f$ represent a direct ancestor of x then

$$\begin{aligned}
DOI(x|f) &< DOI(c_f|f) \\
-(c-1)D(x, r) - cD(x, f) &< -(c-1)D(c_f, r) - cD(c_f, f) \\
-c[D(x, r) + D(x, f) - D(c_f, r) - 1] &< D(c_f, r) - D(x, r)
\end{aligned}$$

$$\begin{aligned}
& -c[D(x, P) + D(P, r) + D(x, P) + \\
& D(P, f) - D(c_f, f) - D(f, P) - D(P, r) - 1] < D(c_f, P) + D(P, r) - D(x, P) - D(P, r) \\
& -c[2D(x, P) - 2] < D(c_f, P) - D(x, P)
\end{aligned}$$

As $D(c_f, P) \geq 2$ — otherwise x would be a child of the focus or the focus itself — we have:

$$0 < D(c_f, P) - 1 \geq 1, \text{ if } D(x, P) = 1.$$

and

$$-c \leq -1 < \frac{D(c_f, P) - D(x, P)}{2D(x, P) - 2}, D(x, P) > 1.$$

Minimising $\frac{D(c_f, P) - D(x, P)}{2D(x, P) - 2}$ implies $D(c_f, P) = 2$, so we have $\frac{2 - D(x, P)}{2D(x, P) - 2}$. Since $0 \leq \left| \frac{2 - D(x, P)}{2D(x, P) - 2} \right| < 1$ we have our proof. \square

To show that the root path nodes have higher DOI values than any other nodes except the focus, it only has to be shown that they have a higher DOI value than the children of the focus, from lemma 1.

Let $P \in PN(x, r) - f$ then:

$$\begin{aligned}
DOI(c_f|f) & < DOI(P|f) \\
-(c-1)D(c_f, r) - cD(c_f, f) & < -(c-1)D(P, r) - cD(P, f) \\
-c[D(c_f, r) + 1 - D(P, r) - D(P, f)] & < D(P, r) - D(c_f, r) \\
-c[D(c_f, f) + D(f, P) + D(P, r) + 1 - D(P, r) - D(P, f)] & < D(P, r) - D(c_f, P) - D(P, r) \\
-2c & < D(c_f, P) \\
c & > \frac{D(c_f, P)}{2} = \frac{D(f, P) + 1}{2}
\end{aligned}$$

Now, as the root path node that is the furthest from the focus is the root, we let $c = \left\lceil \frac{D(r, f)}{2} \right\rceil + 1$ which satisfies $DOI(c_f|f) < DOI(P|f)$. \square

Appendix B

A Tree Analogous All Pairs Shortest Paths Algorithm

This algorithm is adapted from Floyds all pairs shortest algorithm and has a complexity of $O(n^3)$. The algorithm is in C-pseudocode [15]:

```
/* from and to are adjacency matrices,
 * indicating the child->parent and
 * parent->child relationship respectively
 */

/* if from[i][j] = 1, (j,i) is an edge
 * of the graph
 */
int from[N][N];
/* if to[i][j] = 1, (i,j) is an edge of
 * the graph
 */
int to[N][N];

/* initialise from and to */

for (k = 0; k < N; k++) {
    for (i = 0; i < N; i++) {
        for (j = 0; j < N; j++) {
            /* j->k + k->i */
            m = from[k][j] + from[i][k];
            if (m < from[i][j]) {
                from[i][j] = m;
            }
            /* i->k + k->j */
            m = to[i][k] + to[k][j];
            if (m < to[i][j]) {
                to[i][j] = m;
            }
        }
    }
}
```

```

for (k = 0; k <= N; k++) {
  for (i = 0; i <= N; i++) {
    for (j = 0; j <= N; j++) {
      /* k->i + k->j */
      m = from[i][k] + to[k][j];
      if (m < from[i][j]) {
        from[i][j] = m;
      }
    }
  }
}
/* Postcondition:
 *   from contains the tree analogous
 *   all pairs shortest paths
 */

```

Appendix C

The Directory Outline

The outline extracted for the example directory is given [here](#). The format is:

- `dir`, represents a chunk of subdirectories,
- `link`, represents a chunk of static links,
- `no_ext`, represents files with no extension in the current directory,
- any leaf node represents a actual file on the system,
- and any other inner level term is a chunking by extension.

Proceeding a “.” before `dir`, `link` and `no_ext` is preferable to prevent extension clashes.

Davids	t2
link	test
graphlet@	test2
2.1.4/	testa2
dir	testc
private/	testd
dir	tex
rsa/	rsa
no_ext	a
Makefile	eps
a	authent
a2	secure
core	fig
dat	authent
d2	secure
dat2	out
div	gmon
f1	aux
out	rsa
outa	a
outa2	bbl
pro	rsa
profile	blg
rsa2	rsa
rsa*	c
srtd	t
t*	rsa

log	lib\
a	link
rsa	Davids@
dvi	bin\
rsa	Davids\
a	no_ext
nw	Calendar/
a	dir
rsa	xy1997
ps	Mail/
a	no_ext
rsa	Inbox
toc	dir
rsa	drafts
dec	inbox
test	bin/
enc	no_ext
test	java/
a	java
sh	HelloWorld
a	pcfs/
test	no_ext
test2	project/
5/	gz
gz	proj.tar
5.tar	public/
13/	gz
gz	407.tar
13.tar	cosc411.tar
9/	public_html/
gz	html
9.tar	rsa/
C/	noext
RCS/	##
bib/	#test1#
bib	123
davids	Ps
limits/	a1
7/	a
gz	a2
7.tar	div
survival/	header
dir	inv
dates/	main
talk/	mult
assess/	out
tcl/	out1
tcl	profile
1	out2
2	test1
tk/	test2
dir	r
2.1.4	out
dir	gmon

nw	nw,v
integer	rsa
interface	interface
mult	integer
rsa	tarpit/
rsa2	gz
c	graphlet.tar
rsa	tkgcv.tar
log	temp/
rsa	c
tex	test
rsa	no_ext
dir	INBOX
RCS/	